Privacy Protection for P2P Publish-Subscribe Networks

Marek Klonowski, Mirosław Kutyłowski, Bartek Różański

SPI’2005

Wrocław University of Technology

research supported by European Commission, DELIS project
Information systems in Web

- WWW
- listservers, newsgroups and so
- P2P
- Publish-Subscribe (Pub-Sub networks)
Problems

- information monopoly
- spam
- privacy protection
- costs of information retrieval
Groups of common interest

client-server:
- newsgroups/foras: users join a group
- a common network location(s) used to store shared information
- data delivered on user’s request
- drawbacks: non-scalable, subject to spam

Pub-Sub:
- users precisely define contents of their interest
- in a case of an event, all interested subscribers are informed,
- data delivered immediately
- advantages: flexibility, scalability, no unrelated information delivered
Publish-Subscribe

**subscription** precise description of the topic of interest – a virtual group for a combination of topics created

**event** arrival of a new data that matches certain description

**event resolution** the event is associated with subscribers by the Pub-Sub system

**subscriber list** the list of subscribers is forwarded to the server that initiated the event

**delivery** event data is sent to the subscribers by the server that initiated the event
Publish-Subscribe

Important points:

- Pub-Sub is not a routing system,
- P2P based system,
Example Applications

- monitoring changes in the tax system,
- public administration - monitoring changes of regulations concerning a small competence area,
- running a very specific technical system – finding technical support information
Anonymity Problems in Pub-Sub

easy attack violating user’s privacy:

▶ in order to learn who is interested in topic $X$, generate an event on $X$

▶ the system returns automatically the list of all subscribers interested in $X$

▶ it is legal!
Our Goal

- protect user’s privacy
- retain advantages of Pub-Sub
Universal Re-Encryption 1/2

- a message can be re-encrypted by anybody without decryption,
- universal re-encryption does not require knowledge of any key – the ciphertext alone is enough,
Universal Re-Encryption 1/2

- a message can be re-encrypted by anybody without decryption,
- universal re-encryption does not require knowledge of any key – the ciphertext alone is enough,
- it is infeasible to decide whether two ciphertext were encrypted using the same key
- it is infeasible to decide whether ciphertext \( B \) was obtained from ciphertext \( A \) through re-encryption,
Universal Re-Encryption 1/2

- A message can be re-encrypted by anybody without decryption,
- Universal re-encryption does not require knowledge of any key – the ciphertext alone is enough,
- It is infeasible to decide whether two ciphertext were encrypted using the same key,
- It is infeasible to decide whether ciphertext $B$ was obtained from ciphertext $A$ through re-encryption,
- One can compute a ciphertext of $m \cdot m'$ given ciphertexts of $m$ and $m'$

Special case: $m = 1$
Universal Re-Encryption 2/2

Extensions:
  - decryption must be performed by multiple parties,
  - URE signature:
    - over a ciphertext
    - it can be re-encrypted together with the ciphertext

useful to confirming source of a ciphertext in anonymous communication
Anonymous communication with URE-onions

- a random “path” of intermediate nodes is chosen
- message is encoded as a block of URE-ciphertexts, so that:
  - it **must** be processed through the path
    (otherwise it cannot be read)
  - inputs and outputs of an intermediate node **cannot** be linked - universal re-encryption
Navigators

- a URE-onion contains:
  - ciphertexts used for routing
  - ciphertext(s) holding the payload data
- a block devoted for holding an URE-ciphertext (navigator
  cipherbox) contains a ciphertext of 1,
- a message can be inserted into this cipherbox,
- thanks to re-encryption, a navigator can be used many times without security risk
Our protocol

Procedures:

- **subscribing** users inform system about their interest in precisely defined topic
- **recoding** the system recodes user subscription to hide correlations between users and topics from the adversary
- **unsubscribing** users inform Pub-Sub system that they no longer want new data on some topic
- **event handling** upon arrival of some new information users who subscribed to its topic should receive it: ..., preparing routing information, ...
Subscribing

- subscription topic is defined by some predicates: (key, value)-pairs
- subscription request is sent to an appropriate node of Pub-Sub network (P2P routing)
- subscription request contains a navigator and a random ID instead of an address,
- subscription is verified and confirmed,
Recording

**FSL**  Full Subscription List, store all records of user subscriptions (navigators, random IDs)

**RSL**  Reduced Subscription List, are those which are returned upon event arrival – a list of navigators, re-coded each time, some further manipulations (changing the paths)
Event processing

- some event (message) matching predicate $A$ occurs at node $X$
- information about it is sent to P2P server $S$ responsible for $A$
- $S$ replies with a valid RSL list of subscribers
- event message is transmitted anonymously to the subscribers - event message inserted into the navigators,
- spam protection:
  - (option 1) URE- signatures
  - (option 2) some test entries added to RSL (used to monitor the event authors)
Subscriber privacy

- **Subscribing**
  - no addresses provided, only navigators,
  - user preference analysis is more difficult – subscription for different topics with re-encrypted navigators,
  - dummy users prevent data leakage in networks with little dynamics

- **Event handling**
  - if many events on the same A appear, they will be processed (roughly) at the same time posing threat to user anonymity
  - on-line navigators help alleviate this problem - the anonymity paths can be created on-the-fly,
  - traffic analysis futile if anonymity paths have logarithmic length
Protection against spam

- P2P node responsible for the event controls the event message $M$, and provides signed entries of RSL with $M$,
- intermediate path nodes can check URE signature without seeing $M$,
- a message must be dropped if the signature is invalid,
- there is still a problem with repetitions of legitimate messages but Pub-Sub system may generate keys with limited time validity.
Summary

- Pub-Sub protocol with anonymity of subscribers
- Personal data protection acts - fulfilled!
- Higher computational complexity
- Larger communication volume
- Increased communication latency
  but this can be accepted in P2P networks!
- Protocol resistant to malicious nodes
- No trust to nodes assumed/required
- Protection against spam